CSCI3230 / ESTR3108: Fundamentals of Artificial Intelligence 2025-2026 Term 1
The Chinese University of Hong Kong

Assignment 3

Due date: 15 November 2025 (Saturday) 23:59 Full mark: 100
Expected time spent: 5-7 hours

Aims: Understand basic knowledge about neural networks and hands-on programming of simple neural networks, including training and testing.

Description:

In Assignment 3, you will practice on essentials of neural network, including neural network construction, backpropagation, model training and evaluation, PyTorch programming, etc.

Questions:

1. Transformer Concept: Please consider the following statements. Do you think these statements are True or False? If you think it is True, please give the reason; if you think it is False, please give the correct statement. (20%)

(1) A Transformer network processes sentences sequentially from left to right, handling one word at a time.

(2) The primary advantage of multi-head attention is that it significantly reduces the overall computational complexity (from $O(n^2)$ to $O(n^2/h)$, where h is the number of heads) by computing multiple attention heads in parallel.

(3) Statement: In a standard CNN, using 1×1 convolution enables nonlinear feature fusion across channels without altering the spatial dimensions of the feature map.

(4) The feedforward neural network in each Transformer layer operates independently for each position and has a complexity of $O(n^2)$.

2. Consider a simple feedforward neural network with the following architecture:
   - One input Layer: 2 neurons
   - One hidden Layer: 2 neurons with a ReLU activation function
   - One output Layer: 1 neuron with a Sigmoid activation function

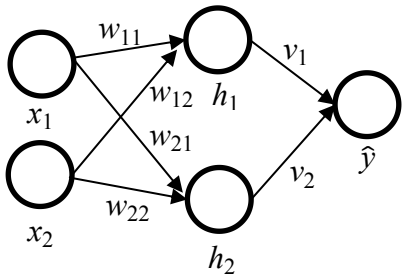
   Weights and Biases:

   Let the weights between the input layer and the hidden layer be:
   - w_{11}, w_{12}
   - w_{21}, w_{22}

   Let the weights between the hidden layer and the output layer be:
   - v_1, v_2

   Let the biases be:
   - Hidden layer biases: b_1, b_2
   - Output layer bias: b_o

   The network structure is shown below, in which h_1, h_2 are the hidden features, and $\hat{y}$ is the final output:



For a given input $[x_1, x_2]$, perform the following calculations. (25%)

(1) Forward Pass: Compute the output $\hat{y}$ of network. (Include activation in your calculations. Use σ to represent the Sigmoid function). (10%)

(2) Loss Calculation: Assume the target output as y. Use mean squared error (MSE) to calculate the loss L. (hint: use y to present the label and $\hat{y}$ to present the output of the network) (5%)

(3) Back propagation:

(3.1) Compute the gradients of the loss with respect to the output layer weights and biases. (hint: compute $\frac{\partial L}{\partial v_1}, \frac{\partial L}{\partial v_2}, \frac{\partial L}{\partial b_o}$. Remove constant coefficients in calculation for simplicity) (5%)

(3.2) Compute the gradients of the loss with respect to the hidden layer weights w_{11}, w_{12}, w_{21}, w_{22} and biases b_1, b_2. (5%)

3. Consider a Convolutional Neural Network (CNN) for binary classification as the following: The goal of this task is to build a CNN that can accurately classify images of cats and dogs. The model need to learn about how to differentiate between the two categories based on the visual features present in the images.

Dataset

You are provided with a dataset organized into two folders for both training and testing datasets (download from link: https://drive.google.com/file/d/181sBYBP_J9Z3S7PDvaIXQiFC-xKmOUhE/view?usp=sharing):

- **cat/**: Contains images of cats, named as "1.jpg", "2.jpg", etc.
- **dog/**: Contains images of dogs, named similarly.

We have prepared the **q3_code.py** to help you to build the model and complete the training. You are required to write the following code (**Finish the q3_code.py. You can change any hyper-parameters and parameters to improve performance.**):

(1) **Model Architecture**: (For [Task 1] in q3_code.py)

- Implement a CNN with the following architecture:
- Convolutional layers followed by ReLU activations and max pooling.
- Fully connected layers with a final output layer using a Sigmoid activation function for binary classification.

(2) **Training**: (For [Task 2] in q3_code.py)

- Train the model using binary cross-entropy loss and the Adam optimizer.
- Monitor the training loss and adjust hyper-parameters as needed.

(3) **Evaluation:** (For [Task 3] in q3_code.py)

Evaluation on test set for each epoch.

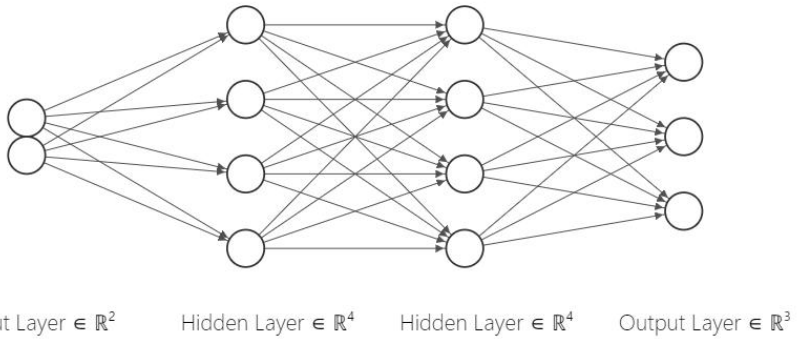
Once you have completed the above code, you should provide the following results. (30%)

(1) **First, plot how the training loss changes during training stage. Explain how you design your training strategies, including key hyper-parameters. Training loss should not fluctuate drastically and should show a downward trend.** (hint: In order to plot the loss, you should record the loss during the training, and provide the graph.) (10%)

(2) **Plot the test accuracy for each epoch and report the final accuracy. Try to achieve a testing accuracy greater than 75%. (10%)**

(3) Finally, do the following things: **1. Try to analyse your training result** (Possible angles: is it a good or bad result? why? how to improve? failure case? influence of the hyper-parameters? overfitting?) **2. Then, try at least 1 more method to further improve the performance and report** (e.g., change the learning rate, add suitable data augmentation). (You can choose multiple angles to answer the question, as long as one viewpoint is reasonable, you will receive full marks. If your attempts to improve the final performance fail, you can still receive full marks as long as you give a reasonable explanation for this). (10%)

4. In this question, we consider a multi-class classification problem, i.e., to discriminate C ($C \geq 3$) classes. We use *Softmax* function in the output layer to produce probability predictions for each class. In specific, consider the following neural network with 2 input neurons (see the figure below), 2 hidden layers and 3 output neurons. Each hidden layer has 4 neurons. For each hidden neuron, we use *ReLU* activation function. The input and output of this neural network is represented by $\boldsymbol{x} \in R^2$ and $\hat{\boldsymbol{y}} \in R^3$. The weight matrix from the input layer to the first hidden layer is denoted by $W_1 \in R^{2\times4}$. The weight matrix from the first hidden layer to the second hidden layer is denoted by $W_2 \in R^{4\times4}$ and the weight matrix from the second hidden layer to the output layer is denoted by $W_3 \in R^{4\times3}$.



Here, we use one-hot encoding to convert ground truth labels into "one-hot" vectors, where only an entry is 1 and other entries are 0. For example, suppose there are C classes in total, the one-hot encoding of a label c (an integer) is a n-dimensional vector where only the c-th entry is equal to 1 while other entries are 0. We use loss function of cross entropy loss to optimize the network. Given the ground truth label in the one-hot encoding format $y=[y_1,y_2,y_3]$ and the probability prediction of the network $p=softmax([\hat{y}_1, \hat{y}_2, \hat{y}_3])=[p_1,p_2, p_3]$, the cross entropy loss between the ground truth label and probability prediction is defined as $l=-\sum_{i=1}^{3} y_i log p_i$. Now solve the following tasks. (25%)

(1) Given a training set and test set listed in *train_q4.csv* and *test_q4.csv*, write PyTorch code to learn the above network. Please randomly initialize the weights matrix with torch.nn.init_kaiming_uniform. Please use the torch.optim.SGD optimizer for training, tune

the learning rate from set {0.4, 0.1, 0.001}, set the batch size as 256 and the total training epochs as 100. Also, fix the seed using code as following:

```
import torch
import torch.backends.cudnn as cudnn
import random
import numpy as np
seed = 1443
cudnn.benchmark = False
cudnn.deterministic = True
random.seed(seed)
np.random.seed(seed)
torch.manual_seed(seed)
torch.cuda.manual_seed(seed)
```

Present the followings: 1) plot the curve of training loss, 2) test the network on the given test set and report the testing accuracy. What's your findings according to these training loss curves and testing accuracy with different learning rates? (20%)

(2) Let's further study the impact of weight initialization. Torch.nn.init provides multiple methods for weight initialization. Please try your best to tune the weight initialization schemes to make the test accuracy ≥95%. Please try at least two methods and show your results. For fair evaluation, please use the torch.optim.SGD optimizer for training, set the learning rate as 0.1, the batch size as 256 and the total training epochs as 100. Also, fix the seed using code as following:

```
import torch
import torch.backends.cudnn as cudnn
import random
import numpy as np
seed = 1443
cudnn.benchmark = False
cudnn.deterministic = True
random.seed(seed)
np.random.seed(seed)
torch.manual_seed(seed)
torch.cuda.manual_seed(seed)
```

After that, please present the followings: (1) plot the curve of training loss (2) test the network on the given test set, report and discuss about the test accuracy. (Hint: Find the documentation for torch.nn.init at https://pytorch.org/docs/stable/nn.init.html) (5%)

Submission:

Submit a single file named <ID>_asmt3.pdf, where <ID> is your student ID.
Your file should contain the following header. Contact Professor Dou before submitting the assignment if you have anything unclear about the guidelines on academic honesty.

```
CSCI3230 / ESTR3108 2025-26 First Term Assignment 3

I declare that the assignment here submitted is original except for source
material explicitly acknowledged, and that the same or closely related material
has not been previously submitted for another course. I also acknowledge that I
am aware of University policy and regulations on honesty in academic work, and
of the disciplinary guidelines and procedures applicable to breaches of such
policy and regulations, as contained in the following websites.

University Guideline on Academic Honesty:
http://www.cuhk.edu.hk/policy/academichonesty/
Faculty of Engineering Guidelines to Academic Honesty:
http://www.erg.cuhk.edu.hk/erg-intra/upload/documents/ENGG_Discipline.pdf

Student Name: <fill in your name>
Student ID  : <fill in your ID>
```

Submit your files using the Blackboard online system.

Notes:

1. Remember to submit your assignment by 23:59pm of the due date. We may not accept late submissions.
2. If you submit multiple times, **ONLY** the content and time-stamp of the **latest** one would be considered.

University Guideline for Plagiarism

Please pay attention to the university policy and regulations on honesty in academic work, and the disciplinary guidelines and procedures applicable to breaches of such policy and regulations. Details can be found at http://www.cuhk.edu.hk/policy/academichonesty/. With each assignment, students will be required to submit a statement that they are aware of these policies, regulations, guidelines and procedures.